Logical Deduction for an Integer Programming Problem and Integer Programming (IP) Formulation for a Logical Problem

Anupama Chanda[1*], Bijan Sarkar[2] & R.N. Mukherjee[3]
[1] Production Engineering Department, Jadavpur University, Kolkata, India
[2] Production Engineering Department, Jadavpur University, Kolkata, India
[3] Department of Mathematics (Retd.), Burdwan University, West Bengal, India
e-mails: anupama.chanda@gmail.com; bsarkar@production.jdvu.ac.in; rnm_bu_math@yahoo.co.in

Abstract

The inherent computational problem that may arise in logic has been addressed and solved. A procedure for converting a set of logical statements into an Integer Programming (IP) has been studied. The same result has also been obtained by application of principle of resolution and Davis-Putnam-Loveland (DPL). Similarly the methods of logic are used for solving IP problems. Few examples have been cited to corroborate the propositions.

Keywords: Propositional Calculus, Resolution, 0-1 Integer Programming, Satisfiability.

1. Introduction

In recent years the use of first order propositional calculus has taken central importance in the formulation and solution of problems taken from diverse area such as management science models, artificial intelligence (AI), mathematical programming [15] particularly integer programming (IP). There has been a close association between logic, artificial intelligence (AI) and mathematical programming particularly integer programming (IP). The main objective of this paper is to investigate and unify the topics related to the above three areas. Here it has been shown that results obtained from zero-one Integer Programming methods are same as those obtained from logical methods of Resolution and Davis-Putnam & Loveland (DPL) procedure.

In this section 2 we present some computational problems that arise in logic. In many applications of medical diagnosis, credit scoring, it is desirable to obtain an economical logical formula which can be used in predicting a result from certain input. It is based on

* Corresponding author.

the Quines procedure [11] and [12] which is used to obtain a minimal logical expression for a propositional calculus statement which can be applied in computer circuit design.

In section 3, logic is applied to an IP problem to facilitate in modeling and representation. This was shown in Granot and Hammer [8] that an IP problem can be modeled in Boolean algebra to improve representation.

In section 4 we describe just the opposite method of modeling a logical inference problem using mathematical programming particularly the integer programming. The recent trend of applying quantitative methods proposed by Hooker [9] to problems in propositional logic have lead to fast inference methods. This was explained in [3], [16] and [18] that an inference problem can be formulated as an IP problem and integer programming algorithms based on 'branch and bound' or 'cutting plane' method or some combination of the two [7] can be applied to solve them. Here in this paper integer programming is used to model an inference problem and the solution is obtained using branch and bound technique particularly and the results obtained are same as those obtained from the traditional inference method of Resolution and Davis-Putnam & Loveland (DPL) procedure. Blair, Jeroslow and Lowe [2] in his paper established the computational comparisons and superiority of Branch and Bound over Resolution and its extension. Chandra and Hooker [4] also discuss comparisons. Further Jeroslow and Wang [10] replaced the LP in branch and bound method by a variable fixing heuristic which resulted in a symbolic method much faster than branch and bound. Beaumont [1] took another approach of converting a MIP model into a DNF and then using a branch and bound algorithm to solve the model.

2. Minimum Logical Statement

$$(X_1 \cdot{\sim}X_2 \cdot X_3) \vee (\sim X_1 \cdot X_2 \cdot X_3) \vee (\sim X_1 \cdot X_2 \cdot \sim X_3) \vee (\sim X_1 \cdot \sim X_2 \cdot X_3) \vee (\sim X_1 \cdot \sim X_2 \cdot \sim X_3) \quad (1)$$

Here X_1, X_2 etc. are atomic propositions having truth value T (True) or F (False). X_1 or its negation X_1 are called literal. These literals when connected by '·' becomes clause and these clauses are connected by v (disjunction) to give a disjunctive normal form (DNF). Here · (and), v (or) and ~ (not) are logical connectives used to build compound propositions. In electrical circuit design it is necessary to obtain a minimum logical expression corresponding to (1) which will contain minimum number of literals.

Resolution is applied to sentence in conjunctive normal form (CNF) while Consensus is used in case of disjunctive normal form (DNF). Quine's procedure [11] and [12] uses consensus and subsumation which is applied repeatedly until no further simplification is possible.

Consensus consist in choosing two clauses where one clause contain exactly one literal negated and other clause contain unnegated literal (literals common to both clauses may or may be present however if present must be of same sign).

Consensus of $X_1 \cdot{\sim}X_2 \cdot X_3$ and $\sim X_1 \cdot{\sim}X_2 \cdot X_3$ is $\sim X_2 \cdot X_3$

Here the common X_1 literal is deleted as it is negated in one clause and unnegated in other. The resultant clause $\sim X_2 \cdot X_3$ is called the resolvent is appended.

In Subsumation one clause contains all the literals of other clause with same sign. The latter clause is said to subsume the former. Here $\sim X_2 \cdot X_3$ subsumes $X_1 \cdot \sim X_2 \cdot X_3$ and is deleted.

Applying consensus and subsumation alternatively we obtain

$$(\sim X_1) v (\sim X_2 \cdot X_3) \tag{2}$$

The individual clauses are called prime implicants (they represent smallest clauses containing least number of literals). The statement (2) is implied by the original statement (1).

3. Logical Deduction for an integer programming (IP) problem

The use of Boolean functions in 0-1 programming was shown by Granot and Hammer [8]. Let us consider an IP constraint:

$$3x_1 + x_2 + x_3 + 2x_4 \leq 3 \qquad x_i \varepsilon \{0,1\} \tag{3}$$

We consider each of the impossibilities of the IP variables simultaneously taking the value 1 and find all such combinations. This gives

$$(X_1 \cdot X_2) \text{ v } (X_1 \cdot X_3) \text{ v } (X_1 \cdot X_4) \text{ v } (X_2 \cdot X_3 \cdot X_4) \tag{4}$$

Where X_i is a proposition and x_i is an integer variable. This statement is the resolvent of the constraint (3). The resolvent of all the constraints are joined by disjunction to give the resolvent of an IP model.

We apply De Morgans Law to obtain the expression (4) in DNF

$$(\sim X_1 \text{ v } \sim X_2) \cdot (\sim X_1 \text{ v } \sim X_3) \cdot (\sim X_1 \cdot \sim X_4) \cdot (\sim X_2 \text{ v } \sim X_3 \text{ v } \sim X_4)$$

4. Integer programming (IP) formulation for a logical problem

It is well known that as the size of logical inference problem increases, it faces exponential complexity and even the traditional inference methods of logic eg. Resolution and its extension fail to solve such problems. This has created great interest in developing a computational procedure which will lead to fast inference method. HP William [17] showed that logical inference problem could be formulated as IP problem and integer programming algorithms particularly 'branch and bound' can be applied to solve them. Let us first take an example and solve it by Resolution method. Later the same logical problem is solved as a satisfiability problem using Davis-Putnam- Loveland (DPL) procedure.

4.1 Resolution

Example1. Is the following deduction valid? [5]

Premises: (i) $X_1 \rightarrow X_2$

(ii) $X_3 \rightarrow X_4$

(iii) $(X_2 \cdot X_4) \rightarrow X_5$

(iv) $\sim X_5$

Conclusion (v) $\sim X_1 \text{ v} \sim X_3$

The resolution given by [13] and [14] is applied to sentence in conjunctive normal form (CNF) while Consensus is used in case of disjunctive normal form (DNF). The premises and negated conclusion are expressed as disjunctive clauses as:

Premises: (i) $\sim X_1 \text{ v } X_2$

(ii) $\sim X_3 \text{ v} X_4$

(iii) $\sim X_2 \text{v} \sim X_4 \text{ v} X_5$

(iv) $\sim X_5$

Negated conclusion: (v) $X_1 \text{ v } X_3$

To show that a propositional statement is valid, resolution assumes that negation of the statement is true and produces contradiction. This in turn proves that the statement is valid. This procedure is demonstrated in the example below.

Here we assume that negation of the conclusion is true ie $X_1 \text{ v } X_3$ is true. We find that there is no way for all of these clauses to be true in a single interpretation and produces contradiction. Hence the conclusion i.e. $\sim X_1 \text{ v} \sim X_3$ is true.

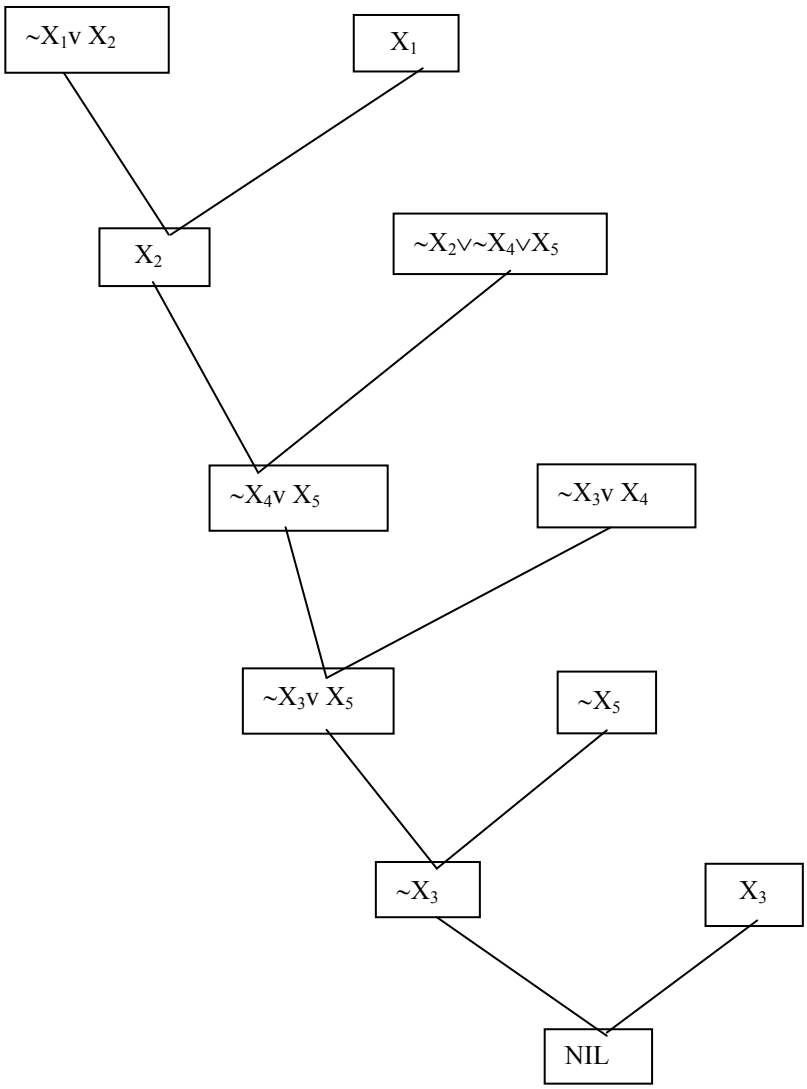

Figure 1: Resolution Tree

4.2 Branch and bound algorithm

The problem will be reformulated as an IP problem suggested by Williams [19] and it would be shown that the results obtained would be same as that obtained by application of resolution principle.

The four premises are expressed as IP constraint

$$\text{(i)} -x_1+x_2 \geq 0 \qquad (3)$$

(ii) $-x_3+x_4 \geq 0$ (4)

(iii) $-x_2-x_4+x_5 \geq -1$ (5)

(iv) $-x_5 \geq 0$ (6)

The conclusion written as an IP

(v) $-x_1 - x_3 \geq -1$ (7)

To show that the deduction is valid, we minimize the conclusion (7) subject to the set of constraints (3) – (6). We use Branch and Bound algorithm with LP relaxation gives the solution

$x_1 = 1 \quad x_2 = 1 \quad x_3 = 0 \quad x_4 = 0 \quad x_5 = 0$ Objective Value (Min) = -1 (8)

This implies that the conclusion (7) is true. Thus it reinforces the same result which is obtained as in resolution method.

4.3 The Davis- Putnam –Loveland (DPL) procedure

DPL [6] is used for testing satisfiability of a logical problem in CNF. A formula of propositional logic is said to be satisfiable if logical values can be assigned to its variables that will make the entire expression true. There are three rules to obtain a satisfying set of truth values:-

a) If a logical expression contain a unit clause i.e. a clause containing single literal X then set X=T. Similarly if a clause consist of a single literal X set X=T.

b) If a logical expression contain a pure literal say it occurs unnegated X in all clauses, set X=T and delete all clauses containing this literal from the CNF. (There may be alternatively satisfying setting but this is sufficient if we are content only to find a satisfiable setting.) Similarly if a literal is negated in all clauses within which it occurs, set~ X =T and delete all clauses containing this literal. (Again there may be alternatives).

c) If a literal is unnegated in some clauses and negated in others then partition the clauses into two sets

   (i) Set X=T and delete the clauses in which it is unnegated.

   (ii) Set ~X= T i.e. X=F and delete the clauses in which it is negated.

Let us consider the same example to understand this.

$(\sim X_1 \vee X_2) \cdot (\sim X_3 \vee X_4) \cdot (\sim X_2 \vee \sim X_4 \vee X_5) \cdot (\sim X_5) \cdot (X_1 \vee X_3)$ (9)

Here $A \rightarrow B$ is obtained by checking satisfiability of $A \cdot \sim B$.

(a) Set $\sim X_5 = T$.

(b) Does not apply.

(c) Branch on X_2 say,

Let $X_2 = T$

(i) $X_4 = F$, $X_3 = F$ and $X_1 = T$

Let $X_2 = F$

(ii) $X_1 = F$, $X_3 = T$ and $X_4 = T$

We obtain the alternative sets of satisfying truth values

$$X_1 = T,\ X_2 = T,\ X_3 = F,\ X_4 = F,\ X_5 = F. \tag{10}$$

$$X_1 = F,\ X_2 = F,\ X_3 = T,\ X_4 = T,\ X_5 = F. \tag{11}$$

Thus one of two solution ie (10) obtained by DPL procedure is same as those obtained from Branch and Bound method of solving Integer Programming (8).

5. Conclusion

In this paper the important connectives with logic, AI and mathematical programming have been reviewed. All the logical problems can be converted into CNF form and subsequently formulated into zero-one integer programming. It is easy to handle and the results obtained from Integer Programming are same as those obtained from logical methods like Resolution and Davis-Putnam & Loveland (DPL) procedure. Also it is shown how the methods of logic can be used to model and solve IP problem.

References

[1] Beaumont, N., 1987, *An algorithm for disjunctive program,* Monash University, Victoria, Australia, 31-68.

[2] Blair, C.E., Jeroslow, R.G. and Lowe, J.K., 1998, Some results and experiments in programming techniques for propositional logic, *Computers and Operations Research*.**13**, 633-645.

[3] Cavalier Tom M., Pardalos Panos M. and Soyster Allen L., 1990, Modeling an Integer , Programming Techniques Applied to Propositional calculus, *Computers Opns Res*.,Vol. 17, No. 6, 561-570.

[4] Chandra, V., and Hooker, J.N., 1988, *Logical Inference: A mathematical programming perspective*, CC- 88-24, School of Industrial Engineering, Purdue University, USA.

[5] Copi Irvin M., 2006, *Symbolic Logic,* published by Printice Hall of India.

[6] Davis M, Putnam H., 1960, A computing procedure for quantification theory, *Journal of the ACM* . **7,** 201-215.

[7] Dowling, W. F. and Gallier, J.H., 1984, Linear time algorithms for testing satisfiability and horn formulae, *Journal of Logic Programming*. **3**, 267-284.

[8] Garfinkel, R.S., and G.L. Nemhauser, 1972, *Integer Programming*, Wiley, New York.

[9] Granot, F., and Hammer, P.L., 1971, On the use of the Boolean functions in 0-1 Programming, *Methods of Operations Research*, **12** 154-184.

[10] Hooker J.N., 1988, A Quantitative Approach to Logical Inference, *Decision Support Systems.* **4**, 45-69.

[11] Jeroslow, R.G., Wang, J., 1987, *Solving proposition satisfiability problems*, working paper, Georgia Institute of Technology, Atlanta, CA.

[12] Quine, W.V., 1952, The problem of simplifying truth functions, *American Mathematical Monthly,* **59**, 521-531.

[13] Quine, W.V., 1955, A way to simplify truth functions, *American Mathematical Monthly*, **62**, 627-631.

[14] Robinson, J.A., 1965, A machine oriented logic based on the resolution principles, *Journal of the ACM*, **12**, 23-41.

[15] Robinson, J.A., 1968, A generalized resolution principle, *Machine intelligence* **3**, 77-93.

[16] Taha Hamdy A., 2007, *Operations Research, An Introduction*, Pearson Education.

[17] Williams H.P., 1977, *Logical problems and integer programming*, Bulletin of the IMA, **13**, 18-20.

[18] Williams H.P., 2009 *Logic and Integer Programming*, published by Springer.

[19] Williams H.P., 1987, Linear and integer programming applied to the propositional calculus, *International Journal of Systems Research and Information Science*, **2**, 81- 100.

[20] Williams H.P., 1995, Logic applied to integer programming and integer programming applied to logic, *European Journal of Operational Research,* **81**, 605-616.